UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/772,392	02/06/2004	Jong-Ky Lee	P56960	5019

7590 10/12/2007

Robert E. Bushnell
Suite 300
1522 K Street, N.W.
Washington, DC 20005

EXAMINER	
ZHANG, SHIRLEY X	

ART UNIT	PAPER NUMBER
4121	

MAIL DATE	DELIVERY MODE
10/12/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

	Application No.	Applicant(s)	
Office Action Summary	10/772,392	LEE, JONG-KY	
	Examiner	Art Unit	
	Shirley X. Zhang	4121	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *02/06/2004*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-18* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-18* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☒ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☒ All b)☐ Some * c)☐ None of:

1.☒ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)

2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)

3) ☒ Information Disclosure Statement(s) (PTO/SB/08)
Paper No(s)/Mail Date _____.

4) ☐ Interview Summary (PTO-413)
Paper No(s)/Mail Date. _____ .

5) ☐ Notice of Informal Patent Application

6) ☐ Other: _____.

DETAILED ACTION

This non-final office action is responsive to the application No. 10/772,392 filed on 02/06/2004.

Priority Claims

Acknowledgement is made of a claim for foreign priority under 35 U.S.C. 119(a)-(d) to the foreign application No. 10-2003-8926 filed in Korea on 02/12/2003.

Information Disclosure Statement

The information disclosure statement (IDS) submitted on 02/06/2004 was filed with the original application. The submission is in compliance with the provisions of 37 CFR 1.97. Accordingly, the examiner is considering the information disclosure statement.

Claim Rejections - 35 USC § 101

35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

1. **Claims 1-3** are rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter.

Claim 1 recites "a lightweight alarm manager running in a Web browser comprising a hearder frame, a data frame and a contents frame" which implies that it is claiming a computer software program that does not fall into any one of the statutory categories under 35 U.S.C. 101, i.e. process, machine, manufacture, composition of matter, or the improvement thereof. Therefore, the subject matter recited in claim one is considered non-statutory subject matter.

Claims 2 and 3 are dependent on claim 1, and further describe the computer software program therefore inherit the 35 U.S.C. 101 issue of the independent claim.

Claim Rejections - 35 USC § 103

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

The factual inquiries set forth in *Graham* v. *John Deere Co.*, 383 U.S. 1, 148 USPQ 459 (1966), that are applied for establishing a background for determining obviousness under 35 39U.S.C. 103(a) are summarized as follows:

1. Determining the scope and contents of the prior art.
2. Ascertaining the differences between the prior art and the claims at issue.
3. Resolving the level of ordinary skill in the pertinent art.
4. Considering objective evidence present in the application indicating obviousness or nonobviousness.

2. **Claims 1, 3 and 10-13** are rejected under 35 U.S.C. 103(a) as being unpatentable over Ditmer et al. (U.S. Patent 6,473,407, hereinafter "**Ditmer**"), in view of Dorland et al. (U.S. Application Publication 2003/0028577, hereinafter, "**Dorland**").

As to claim 1, Ditmer teaches a lightweight alarm manager running in a Web browser (column 15, lines 50-57 disclose a browser-based event monitor that is launched from the home page shown in Fig.4, therefore the event monitor is a lightweight alarm manager) to be applied to a computer connected to NMS (Network Management System) over network, comprising:

a header frame fixing a title label of the alarm manager (Fig. 4 discloses a header frame that displays the title label "networkMCI Interact");

a data frame receiving alarm information from the NMS through the network (column 16, lines 65-67 and column 17, lines 1-3 disclose that an alarm management object is launched by the GUI client application to handle communications with the multi-tiered server for events or alarms, therefore the alarm management object is equivalent to the data frame recited in the claim. Note that the multi-tiered server includes the Web server/dispatcher/cookie jar server 635, event monitor proxy 640 and the event monitor server 650); and

a contents frame having dynamic HTML (Hypertext Markup Language) for reading the alarm information being managed in the data frame (column 16, lines 23-55 disclose COApp and COApplet as two different embodiments of the event monitor GUI client application; column 16, lines 35-45 further disclose that a Java-based COAppFrame provides displace space for the event monitor; column 16, lines 51-55 disclose in another embodiment that the Java applet COApplet launched by the browser from an HTML <Applet>tag provides the event monitor with a browser-based display space, therefore both COApp and COApplet are equivalent to the contents frame recited in the claim);

Ditmer does not explicitly teach constructing a data table of the alarm information being managed by the data frame. However, Ditmer teaches in column 5, lines 39-43 that the client software includes additional applications that are directed to front-end services such as the presentation of data in the form of tables and charts.

Therefore, it would have been obvious to one of ordinary skill in the art at the time of the invention to have the contents frame construct a data table of the alarm information being managed by the data frame, if the said alarm information is to be presented in the form of tables, as suggested by Ditmer. Such implementation is in the knowledge generally available to one of ordinary skill in the art at the time the invention was made.

Furthermore, Ditmer does not teach that the alarm manager manages the alarm information in XML format.

However, Dorland teaches managing alarm information in XML format by disclosing in paragraph [0028] that the alarm information requested by the web client is retrieved from the database, converted into XML format, and then passed to the web client via the web server.

Therefore, it would have been obvious to one of ordinary skill in the art at the time of the invention to apply Dorland's teaching of managing the alarm information in XML format to Ditmer. At the time the invention was made, one would have been motivated to combine Ditmer and Dorland because XML was then already recognized as a general-purpose markup language that facilitated the sharing of structured data across different information systems, particularly over Internet. Such knowledge is also disclosed in paragraph [0007] of Dorland which stated that it would be desirable to provide a common platform to all client applications to receive the events/data in a universal language, i.e. XML.

As to claim 3, the combination of Ditmer and Dorland teaches the alarm manager of claim 1.

Ditmer further teaches that the contents frame provides the alarm information that is comprised of {severity} of the alarm, {eventtime} when the alarm is raised, alarm ID{alarm_id}, components of network equipment, {dn}, where the alarm is raised, and contents of the alarm {contents} (column 27, lines 25-34 disclose that the attributes of alarms recorded are severityLevel_, timeStamp_, alarmID_, interfaceID_, and alarmText_, which correspond to the {severity}, {eventtime}, {alarm_id}, {dn} and {contents} recited in the claim, respectively).

As to claim 10, Ditmer teaches a method of providing alarm information to a lightweight alarm manager running in a Web browser, the method comprising the steps of:

receiving, at NMS (Network Management System), an alarm information request from an alarm manager via a network (column 21, lines 61-66 disclose that the StarWeb Server, a Web server, receives requests from the client);

confirming, at the NMS, session information related to the alarm manager (column 22, lines 22-25 disclose that after receiving a request, the StarWeb server needs to establish that the request has come from a valid user and then maps the request to its associated session) and obtaining time information used to form an alarm information packet to be transmitted to the alarm manager (column 20, lines 35-45 disclose that the user requesting alarm/event information may apply limitations to the data to be retrieved by including an event view in the request; column 20, lines 59-61 further disclose that "Date and Time Elements" may be specified in the event view to limit the requested data to a certain date and time period. Therefore, the "Date and Time Elements" in event view is equivalent to the time information recited in the claim).

obtaining, at the NMS, additional alarm information from a database in the NMS, the additional information being based on the time information, the additional information being added to the packet (column 22, lines 41-46 disclose that the StarWeb server forwards the client request to the corresponding service proxy via a Dispatcher; column 23, lines 2-17 disclose that the service proxy retrieves from its database the requested information; column 21 lines 22-25 further disclose that the SQL server, i.e. database server, builds a report of event data specified by the event view, which may include the "Date and Time Elements" that limits the event data to a date and time period.);

converting the packet into a format at the NMS, and transmitting the packet of alarm information in the said format to the alarm manager (column 23, lines 14-16 disclose that data returned from "networkMCI Interact"'s server, i.e. the multi-tiered server that includes the Web server, dispatcher, cookie jar server, event monitor proxy and the event monitor server, is translated back to client format, and returned to the client via web server/dispatcher).

Ditmer does not teach but Dorland teaches that the packet to be transmitted to the alarm manager is in XML format. Dorland teaches in paragraph [0028] that the alarm information requested by the web client is retrieved from the database, converted into the XML format, and then passed to the web client via the web server.

It would have been obvious to one of ordinary skill in the art at the time of the invention to apply Dorland's teaching of managing the alarm information in XML format to Ditmer. At the time the invention was made, one would have been motivated to combine Ditmer and Dorland because XML was then already recognized as a general-purpose markup language that facilitated the sharing of structured data across different information systems, particularly over Internet. Such knowledge is also disclosed in paragraph [0007] of Dorland which stated that it would be desirable to provide a common platform to all client applications to receive the events/data in a universal language, i.e. XML.

As to claim 11, the combination of Ditmer and Dorland teaches the method of claim 10.

Ditmer further teaches the step of managing, at the NMS, the session information related to each of the alarm managers via a checkSession thread (column 8, lines 17-30 disclose that the multi-tiered server maintains a logical session for each user and the sessions are initiated and tracked by a "cookie jar server", which implies that the multi-tiered server runs a thread or process to manage user sessions, such a thread or process is equivalent to the checkSession thread recited in the claim).

As to claim 12, Ditmer teaches a method of providing alarm information to a lightweight alarm manager running in a Web browser, the method comprising the steps of:

receiving, at NMS (Network Management System), an alarm information request from an alarm manager via a network (Network Management System), an alarm information request from an alarm manager via a network (column 21, lines 61-66 disclose that the StarWeb server, i.e., a component of the multi-tiered server, receives requests from the client. Note that the multi-tiered server includes the Web server/dispatcher/cookie jar server 635, event monitor proxy 640 and the event monitor server 650);

creating, at the NMS, a service thread for providing alarm information at a request of the alarm manager (column 24, lines 21-23 disclose that the multi-tiered server has multithreaded proxy functionality that allows the proxies to service multiple clients simultaneously; column 25, lines 36-37 further disclose that the client request is processed by the forked child process, which is equivalent to a service thread recited in the claim), and a checkSession thread for managing session information related to the alarm manager (column 8, lines 17-30 disclose that the multi-tiered server maintains a logical session for each client and the sessions are initiated and tracked by a "cookie jar server", which implies that the multi-tiered server runs a thread or process to manage client sessions, such a thread or process is equivalent to the checkSession thread recited in the claim);

determining, at the service thread, whether session information related to the alarm manager is present in the NMS, and when the session information is not present, creating a new session information, and when the session information is present, extracting a final search alarm occurrence time from the session information (column 8, lines 17-30 disclose that the multi-tiered server maintains a logical session for each user and the sessions are initiated and tracked by a "cookie jar server", which implies that the cookie jar server maps user request to a logical session and if the user session does not exist creates a new one);

based on the alarm occurrence time, obtaining, via the service thread, additional alarm information by searching a database in the NMS and updating the session information accordingly with information found in the database, said additional alarm information being based on the alarm occurrence time (column 20, lines 38-44 and lines 59-61 disclose that the user can obtain alarm/event information pertaining to a certain date and time period by specifying the "Date and Time Elements" in the event view, therefore the time period specified in "Date and Time Elements" can be set to that of the alarm occurrence time; column 21 lines 22-25 further disclose that the alarm information as specified by event view is retrieved from a SQL server);

transmitting the alarm information to the alarm manager as a response to the request (column 21, lines 26-27 disclose the event report built by the SQL server is sent to the client browser via the web/dispatch servers); and

checking, at the checkSession thread, an update date of the session info, and deleting the session information when the session information is not valid (column 25, lines 46-53 disclose that a validation step is performed by connecting to the Web server cookie jar to determine if the current session is a valid user session, and close the connection if the current session is invalid. It is implicit in Ditmer's disclosure that the validation step is performed within either a thread or a process that is equivalent to the checkSession thread recited in the claim.).

Ditmer does not teach converting, at the service thread, the alarm information into an XML format.

However, Dorland teaches in paragraph [0028] that the alarm information requested by the web client is retrieved from the database, converted into XML format, and then passed to the web client via the web server.

It would have been obvious to one of ordinary skill in the art at the time of the invention to apply Dorland's teaching to Ditmer such that the alarm information passed from Ditmer's NMS server to the NMS manager would be in XML format. At the time the invention was made, one would have been motivated to combine the teachings from Ditmer and Dorland because XML was then already recognized as a general-purpose markup language that facilitated the sharing of structured data across different information systems, particularly over Internet. Such knowledge is also disclosed in paragraph [0007] of Dorland which stated that it would be desirable to provide a common platform to all client applications to receive the events/data in a universal language, i.e. XML.

As to claim 13, the combination of Ditmer and Dorland teaches the method of claim 12.

Ditmer further teaches that the session information has information about the alarm manager that sent the request, information about a user using the alarm manager (column 8, lines 5-7 disclose that an HTTPS session is designated by a logon, successful authentication of user, followed by use of server resources and logoff, therefore it is inherent that a session has information about the user trying to logon, and information about the event monitor sending the request), and information about time of final alarm occurrence of the alarm manager (column 21, lines 46-50 disclose that if a session remains open, the server will report new events defined by the event view as they are received by the server, which implies that the server must know and store the time of last event occurrence).

3. **Claim 2 is** rejected under 35 U.S.C. 103(a) as being unpatentable over Ditmer and Dorland, as applied to claim 1 above, further in view of Flanagan (Chapter 17 of the book "JavaScript: The definitive Guide, 4th Edition", hereinafter "**Flanagan**")

As to claim 2, the combination of Ditmer and Dorland teaches the alarm manager of claim 1.

Ditmer further teaches a contents frame that provides a GUI (Graphical User Interface) (column 5, lines 53-62 disclose that the client software provides a reusable'and common GUI abstraction).

Furthermore, Ditmer discloses that the client software includes additional applications that are directed to front-end services such as the presentation of data in the form of tables and charts (column 5, lines 39-43).

Neither Ditmer nor the combination of Ditmer and Dorland explicitly teaches a table object of a HTML provided by the Web browser.

However, Flanagan teaches that an HTML table objects is provided by a Web browser to display information (pages 27-29 of chapter 17 disclose the method of displaying information on a Web browser by loading an XML document, extracting information from it, and dynamically creating an HTML version of that information for display in a HTML table object provided by the browser. See page 28, example 17-9 for more details).

Therefore, it would have been obvious to one of ordinary skill in the art at the time of the invention to implement the tables recited by Ditmer as an HTML table object, as taught by Flanagan. One would have been motivated to combine as such for the reason that at the time the invention was made, HTML table object was already a well-defined element for supporting structured display of data such as alarm information on a Web browser, and software tools such as JavaScript were readily available to dynamically edit the table contents.

4. **Claim 4** is rejected under 35 U.S.C. 103(a) as being unpatentable over Ditmer et al. (U.S. Patent 6,473,407, hereinafter "**Ditmer**").

As to claim 4, Ditmer teaches a service method of a lightweight alarm manager running in a Web browser (column 15, lines 50-57 disclose a browser-based event monitor that is launched from the home page shown in Fig.4, therefore the event monitor is a lightweight alarm manager), to be applied to a computer connected NMS (Network Management System) through network, the service method comprising the steps of:

receiving a request from a user to use the alarm manager (column 15, lines 55-57 disclose that the user can request to launch the event monitor by clicking the event monitor icon on the home page, as shown in Fig.4);

creating a header frame, a contents frame, and a data frame on the Web browser in the alarm manager, in response to an alarm manager service request from a user (Fig. 4 discloses a **header frame** that displays the title label "networkMCI Interact"; column 16, lines 65-67 and column 17, lines 1-3 disclose that an alarm management object is launched by the GUI client application to handle communications with the multi-tiered server for events or alarms, therefore the alarm management object is equivalent to the **data frame** in the instant application; column 16, lines 23-55 disclose COApp and COApplet as two different embodiments of the event monitor GUI client application; column 16, lines 35-45 further disclose that a Java-based COAppFrame provides displace space for the event monitor; column 16 lines 51-55 disclose in another embodiment that a Java applet COApplet launched by the browser from an HTML <Applet> tag can provide the event monitor with a browser-based display space; therefore COApp and COApplet are the equivalents of the **contents frame;** Furthermore, all the above applications were launched by clicking the event monitor icon on the home page. Note that the multi-tiered server includes the Web server/dispatcher/cookie jar server 635, event monitor proxy 640 and the event monitor server 650);

requesting, at the data frame, that the NMS provides alarm information periodically to the data frame (column 21, lines 52-58 disclose that reports of events may be periodically forwarded based on a customer configurable interval to the client browser application, i.e. the data frame);

managing the alarm information in the alarm manager when the alarm information is received by the data frame (column 13, lines 28-39 disclose a list of alarm management activities carried out by the event monitor);

periodically checking, by the contents frame made up of dynamic HTML, whether the alarm information in the data frame is being properly managed (column 16 lines 51-55 disclose that a Java applet COApplet launched by the browser from an HTML <Applet> tag can provide the event monitor with a browser-based display space, i.e., the contents frame is made up of dynamic HTML; column 17, lines 3-6 disclose that an alarm thread is created to run periodically to poll for current event monitor alarms);

accessing and obtaining, by the contents frame, the alarm information being managed by the data frame (column 17, lines 7-9 disclose that the said alarm thread creates a command to update the display when it receives data back from the web server, i.e. the contents frame is updated with information managed by the data frame);

displaying the alarm information to the user (column 16, lines 39-44 and column 16, lines 52-57 disclose the Java classes COAppFrame and COApplet that provides display space for the alarm information).

Ditmer does not explicitly teach constructing a data table of the alarm information being managed by the data frame. However, Ditmer teaches in column 5, lines 39-43 that the client software includes additional applications that are directed to front-end services such as the presentation of data in the form of tables and charts.

Therefore, it would have been obvious to one of ordinary skill in the art at the time of the invention to have the contents frame construct a data table of the alarm information being managed by the data frame, if the said alarm information is to be presented in the form of tables, as suggested by Ditmer. Such implementation is in the knowledge generally available to one of ordinary skill in the art at the time the invention was made.

5. **Claim 5** is rejected under 35 U.S.C. 103(a) as being unpatentable over Ditmer as applied to claim 4 above, further in view of Dorland et al. (U.S. Application Publication 2003/0028577, hereinafter, "**Dorland**").

As to claim 5, Ditmer teaches the method of claim 4, wherein the requesting step is comprised of the sub-steps of

requesting the NMS connected to the data frame through the network to provide alarm information periodically (column 17, lines 1-6 disclose that the alarm management object periodically creates an alarm thread to poll for current event monitor alarms);

receiving alarm information from the NMS (column 17, line 7 discloses that the alarm thread receives the data back from the web server); and

managing the received alarm information (column 18, lines 41-45 disclose that the event monitor manages the received alarm information by launching a pre-defined trouble shooting procedure);

Ditmer does not teach that the alarm information is received and managed in XML format.

However, Dorland teaches managing alarm information in XML format by disclosing in paragraph [0028] that the alarm information requested by the web client is retrieved from the database, converted into XML format, and then passed to the web client via the web server.

It would have been obvious to one of ordinary skill in the art at the time of the invention to apply Dorland's teaching of managing the alarm information in XML format to Ditmer. At the time the invention was made, one would have been motivated to combine Ditmer and Dorland because XML was then already recognized as a general-purpose markup language that facilitated the sharing of structured data across different information systems, particularly via the Internet. Such knowledge is also disclosed in paragraph [0007] of Dorland which stated that it would be desirable to provide a common platform to all client applications to receive the events/data in a universal language, i.e. XML.

6. **Claims 6 and 8** are rejected under 35 U.S.C. 103(a) as being unpatentable over Ditmer as applied to claim 4 above, in view of Flanagan ("JavaScript: The definitive Guide, 4th Edition, hereinafter "**Flanagan**").

As to claim 6, Ditmer teaches the method of claim 4, wherein the accessing and obtaining, constructing and displaying steps comprise:

obtaining, by the contents frame, the alarm information in XML format from the NMS (column 17, line 7 discloses that the alarm thread receives the data back from the web server);

managing, by the data frame, the received alarm information in the data frame (column 18, lines 41-45 disclose that the event monitor manages the received alarm information by launching a pre-defined trouble shooting procedure);

Ditmer further discloses in column 5, lines 39-43 that the client software includes additional applications that are directed to front-end services such as the presentation of data in the form of tables and charts.

Ditmer does not teach but Flanagan teaches adding a row to a table object, by the contents frame, using attributes of the table object of a HTML provided by the Web browser; and displaying the alarm information being obtained using the table object (Flanagan, chapter 17, pages 27-29, "The Document Object Model" disclose that an HTML table object can be used to display information extracted from an XML document; Flanagan, pages 28, example 17-9 further discloses that JavaScript supports the DOM object HTMLTableElement, which provides the method HTMLTableElement.insertRow() for adding a row to an HTML table object. The HTML page is automatically reloaded after the HTML table object is populated, causing the new alarm information to be displayed).

Therefore, it would have been obvious to one of ordinary skill in the art at the time of the invention to combine Flanagan with Ditmer such that the contents frame would use attributes of an HTML table object provided by the Web browser to add a row to the table and display the alarm information. One would have been motivated to combine as such for the reason that HTML table object was already a well-defined element for supporting structured display of data such as alarm information on a Web browser, and software tools such as JavaScript were readily available to dynamically edit the table contents, as taught by Flanagan.

As to claim 8, the combination of Ditmer and Flanagan teaches the method of claim 6.

Neither Ditmer nor the combination of Ditmer and Flanagan specifically teaches checking whether a number of current rows in the table object provided by the Web browser is greater than a predetermined number of rows;

However, it would have been obvious to one of ordinary skill in the art at the time of the invention to check whether the number of current rows in the table object was greater than a predetermined number of rows because it was as simple as comparing two integers.

Furthermore, Ditmer does not teach but Flanagan teaches the step for adding a row by the contents frame comprises sub-steps of:

deleting the oldest record when the number of current rows in the table object provided by the Web browser is greater than the predetermined number; creating a new row in the table object comprising the alarm information read by the contents frame; and displaying the alarm information of the table object when the number of current rows in the table object provided by the Web browser is not greater than the predetermined number of rows to be maintained (Flanagan, book chapter 17, "The Document Object Model" disclose that JavaScript supports the DOM object HTMLTableElement that supports the methods HTMLTableElement.deleteRow() and HTMLTableElement.insertRow() for deleting a row from or adding a row to an HTML table object. The HTML page is reloaded after the HTML table object is populated, causing the new alarm information to be displayed).

Therefore, it would have been obvious to one of ordinary skill in the art at the time of the invention to combine Flanagan with Ditmer such that the contents frame would use attributes of an HTML table object provided by the Web browser to delete the oldest record when the number of current rows in the table object provided by the Web browser is greater than the predetermined number, create a new row in the table object comprising the alarm information read by the contents frame, and display the alarm information of the table object when the number of current rows in the table object provided by the Web browser is not greater than the predetermined number of rows to be maintained.

One would have been motivated to combine as such for the reason that HTML table object was already a well-defined element in HTML for supporting structured display of data such as alarm information on a Web browser, and software tools such as JavaScript were readily available to dynamically edit the table contents, as taught by Flanagan.

7. **Claims 7 and 9** are rejected under 35 U.S.C. 103(a) as being unpatentable over Ditmer and Dorland as applied to claim 5 above, further in view of Flanagan ("JavaScript: The definitive Guide, 4th Edition, hereinafter "**Flanagan**").

As to claim 7, the combination of Ditmer and Dorland teaches the method of claim 5.

Ditmer further teaches that the accessing and obtaining; constructing and displaying steps comprise:

obtaining, by the contents frame, the alarm information in XML format from the NMS; managing, by the data frame, the received alarm information in the data frame (column 17, line 7 discloses that the alarm thread receives the data back from the web server);

Neither Ditmer nor the combination of Ditmer and Dorland teaches adding a row to a table object, by the contents frame, using attributes of the table object of a HTML provided by the Web browser; and displaying the alarm information being obtained using the table object.

However, Flanagan teaches in chapter 17 "The Document Object Model" that JavaScript supports the DOM object HTMLTableElement, a table object of a HTML page provided by the Web browser to display information in a structured manner. The HTMLTableElement supports the method HTMLTableElement.insertRow() for adding a row to an HTML table object, and the

HTML page is reloaded after the HTML table object is populated, causing the new alarm information to be displayed.

Therefore, it would have been obvious to one of ordinary skill in the art at the time of the invention to combine Flanagan with Ditmer and Dorland such that the contents frame would use attributes of an HTML table object provided by the Web browser to add a row to the table and display the alarm information. One would have been motivated to combine as such for the reason that HTML table object was already a well-defined element for supporting structured display of data such as alarm information on a Web browser, and software tools such as JavaScript were readily available to dynamically edit the table contents, as taught by Flanagan.

As to claim 9, the combination of Ditmer, Dorland, and Flanagan teaches the method of claim 7.

Neither Ditmer nor the combination of Ditmer, Dorland, and Flanagan specifically teaches checking whether a number of current rows in the table object provided by the Web browser is greater than a predetermined number of rows;

However, it would have been obvious to one of ordinary skill in the art at the time of the invention to check whether the number of current rows in the table object is greater than a predetermined number of rows because this is as simple as comparing two integers.

Flanagan further teaches the step for adding a row by the contents frame comprises sub-steps of:

deleting the oldest record when the number of current rows in the table object provided by the Web browser is greater than the predetermined number; creating a new row in the table object comprising the alarm information read by the contents frame; and displaying the alarm information of the table object when the number of current rows in the table object provided by the Web browser is not greater than the predetermined number of rows to be maintained (Flanagan, book chapter 17, "The Document Object Model" disclose that JavaScript supports the DOM object HTMLTableElement that supports the methods HTMLTableElement.deleteRow() and HTMLTableElement.insertRow() for deleting a row from or adding a row to an HTML table object. The HTML page is reloaded after the HTML table object is populated, causing the new alarm information to be displayed).

Therefore, it would have been obvious to one of ordinary skill in the art at the time of the invention to combine Flanagan with Ditmer such that the contents frame would use attributes of an HTML table object provided by the Web browser to delete the oldest record when the number of current rows in the table object provided by the Web browser is greater than the predetermined number, create a new row in the table object comprising the alarm information read by the contents frame, and display the alarm information of the table object when the number of current rows in the table object provided by the Web browser is not greater than the predetermined number of rows to be maintained.

One would have been motivated to combine as such for the reason that HTML table object was already a well-defined element in HTML for supporting structured display of data such as alarm information on a Web browser, and software tools such as JavaScript were readily available to dynamically edit the table contents, as taught by Flanagan.

8. **Claims 14-18** are rejected under 35 U.S.C. 103(a) as being unpatentable over Ditmer et al. (U.S. Patent 6,473,407, hereinafter "**Ditmer**") in view of Carey et al. ("Heterogeneous Tools for Heterogeneous Network Management with WBEM", hereinafter, "**Carey**").

As to claim 14, Ditmer teaches a NMS (Network Management System) server (Fig. 6 discloses a multi-tiered server that includes Web Servers/Dispatcher 635, event monitor proxy 640 and event monitor server 650. The multi-tiered server is equivalent to a NMS server for the alarm and event monitoring purpose) adapted to serve a plurality of alarm managers (Fig. 6, where the client browser that runs GUI client applications such as Java applets or Java applications is an alarm manager), the NMS server comprising:

an engine adapted to transmit data requested by the user to a data frame of one of said plurality of lightweight alarm managers (column 21, lines 22-27 and column 23, lines 7-17 together disclose that the Server 650 retrieves requested data from the database, and returns the data to the client browser via the web/dispatch servers, which may translate the data to the client format if necessary, therefore server 650 is equivalent to the engine recited in the claim);

A context adapted to store session information about said plurality of lightweight alarm managers (column 8, lines 1-30 disclose a cookie jar server that is dedicated to manage client/server sessions, where a session is identified by a cookie generated by the server and is designated by a logon, successful authentication, use of server resources and logoff); and

a database comprising alarm information (Fig. 8, column 19, lines 66-67 and column 20, lines 1-2 disclose that the above mentioned Server 650 comprises a database 410).

Ditmer does not teach but Carey teaches

a JSP (Java Server Page) engine comprising a makeXML JSP adapted to transmit XML data to a data frame of one of said plurality of lightweight alarm managers (Carey, section 2.2 discloses that XML is employed to represent the management data in a standard way; Carey, section 4 “A web Based Manager” further discloses that Java Server Page (JSP) and JavaBeans are used to develop a web-based heterogeneous network manager that is equivalent to an NMS server);

a JSP context adapted to store session information about said plurality of lightweight alarm managers (Carey, section 4.1 "Developing the Manager Application" discloses that JavaBeans are employed for user validation operations, therefore the JavaBeans are equivalent to a JSP context).

It would have been obvious to one of ordinary skill in the art to combine Ditmer with Carey so that the NMS server employs the JSP engine and JSP context to realize its functionality as disclosed in the claim. One would have been motivated to make such modification because at the time the invention was made, there was a demand and a trend in the art of network management to utilize standard, platform independent technologies such as XML and JSP to manage heterogeneous networks encompassing traditional telecommunication networks, computer communication networks and storage networks, as is taught in the "Abstract" of Carey. The WBEM (Web Based Enterprise Management) initiative is another evidence of this trend.

As to claim 15, the combination of Ditmer and Carey teaches the NMS server of claim 14.

Ditmer further teaches that the server provides a service thread that is adapted to offering alarm information at the request of each of the plurality of lightweight alarm managers (Fig. 12(a) and column 25, lines 36-37 disclose that the multi-tiered server forks a child process to handle the client request, such a child process is equivalent to the service thread) and a checkSession thread adapted to manage (or check) an existence of each of the plurality of lightweight alarm managers (column 25, lines 44-48 disclose that the event monitor makes a connection to the web server's cookie jar to validate the session, which implies that the validation step is performed in a thread equivalent to the checkSession thread.).

As to claim 16, the combination of Ditmer and Carey teaches the NMS server of claim 15.

Ditmer further teaches that the makeXML JSP regularly receives a HTTP request from each of the lightweight managers (Fig. 6 shows that Web Server/Dispatcher receives HTTP requests from the client running in the web browser).

Ditmer does not explicitly teach the step of confirming a final date and time to extract new alarm information from the session information in the JSP Context.

However, Ditmer teaches that the client can specify the "Date and Time Elements" in an event view so that the server will retrieve alarm/event information pertinent to that particular period of time (column 20, lines 40-45 and column 20, lines 59-61). Therefore, the user can set "Date and Time Elements" to the period between the last time the alarm information was retrieved, i.e., the final data and time, and the time the current request is made, and transmit this information to the server as part of the request.

It would have been obvious to one of ordinary skill in the art at the time of the invention to modify Ditmer so that a copy of the final date and time information is also stored in the server so that the server can confirm the final date and time the next time the user sends a request. Such a modification to Ditmer is a matter of design choice and implementation preference.

As to claim 17, the combination of Ditmer and Carey teaches the NMS server of claim 14.

Ditmer further teaches that the makeXML JSP, using a final date and time as a starting point (column 20, lines 40-45 and column 20, lines 59-61 disclose that the client can retrieve alarms that occurred during a certain date and time period by using the "Date and Time Elements" in an event view that is transmitted to the server as part of the request. Therefore, Ditmer can set the "Date and Time Elements" using a final date and time as a starting point. Here the final date and time is interpreted to be the last time the alarm information is retrieved from the server.);

queries data from the database and constructs a document representing the data from the database and transmits the document to a data frame of one of the plurality of lightweight alarm managers (Fig. 9, column 20, lines 35-67 and column 21, lines 19-34 disclose that the client request is forwarded to the SQL database server via intermediate servers, the SQL server then extracts data and builds a report of event data specified by the event view, and sends the report to the client browser.)

Ditmer does not teach but Carey teaches using XML as the document format (section 2.2 "Transport Encoding" teaches transporting common information model (CIM) classes and instances as XML documents).

It would have been obvious to one of ordinary skill in the art at the time of the invention to apply Dorland's teaching of managing the alarm information in XML format to Ditmer. At the time the invention was made, one would have been motivated to combine Ditmer and Carey because XML was then already recognized as a general-purpose markup language that facilitated the sharing of structured data across different information systems, particularly over Internet. Such knowledge is also disclosed in section 2.2 of Carey, which stated that for applications to interoperate with one another it must be possible to represent actual management data in a standard way, i.e. XML.

As to claim 18, the combination of Ditmer and Carey teaches the NMS server of claim 14.

Ditmer further that the JSP context comprising the stored session information is comprised of NMS user information using the plurality of alarm managers (column 8, lines 1-30 disclose that a cookie jar server manages the client/server sessions, whose tasks include user authentication and verification, which implies that the session information must include the user information for authentication purpose) and information about a time when a corresponding one of said plurality of alarm managers raises a final alarm (column 20, lines 40-45 and column 20, lines 59-61 disclose that the client can retrieve alarms that occurred during a certain date and time period by using the “Date and Time Elements” in an event view that is transmitted to the server as part of the request; column 21, lines 46-58 further disclose that if the session remains open, server will periodically report new events defined by the event view; therefore it is inherent that the user session managed by the server contains information about a time when a corresponding one of said plurality of alarm managers raises a final alarm.).

Conclusion

9. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

U.S. Patent 5,987,513, “Network Management Using Browser-Based Technology”;

U.S. Patent 6,308,205, “Browser-based network management allowing administrators to use web browser on user's workstation to view and update configuration of network devices”;

U.S. Patent 6,247,052, "Graphic user interface system for a telecommunications switch management system";

U.S. Patent 6,487,590, "Method for controlling a network element from a remote workstation";

U.S. Patent 6,363,421, "Method for computer internet remote management of a telecommunication network element";

U.S. Patent 6,965,932, "Method and architecture for a dynamically extensible web-based management solution";

U.S. Patent 6,167,448, "Management event notification system using event notification messages written using a markup language";

U.S. Patent Application Publication 2001/0052013, "Integrated interface for web-based customer care and trouble management";

Wren et al., "Agent and Web-based Network Management", IEEE Global Telecommunications Conference (GlobeCom'99), 1999, pages 1877-1881;

IETF RFC 1942, "HTML Tables", 1996

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Shirley X. Zhang whose telephone number is (571) 270-5012. The examiner can normally be reached on Monday through Friday 7:30am - 5:00pm EST.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Taghi Arani can be reached on (571) 272-3787. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

TAGHI ARANI
PRIMARY EXAMINER
10/9/07